Resource Allocation Problem
Using
Machine learning

submitted by

Rahul Kurkure (20114079), Sanchit Chaba (20114083)

for the fulfillment

of

Cloud Computing (CSN-520) Project

Under the guidance of

Prof. Sateesh Kumar Peddoju

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY ROORKEE

ROORKEE-247667

April-May 2024

Chapter-1

1.1 Introduction

In the realm of cloud computing, efficiently allocating resources is crucial for ensuring optimal performance and meeting the diverse needs of users. Resource allocation involves assigning available computing resources, such as virtual machines (VMs) and storage, to various tasks and applications running on cloud infrastructure. However, traditional methods of resource allocation may struggle to adapt to the dynamic and unpredictable nature of cloud workloads. As cloud computing continues to evolve and grow in complexity, the need for more sophisticated and adaptive resource allocation techniques becomes increasingly apparent. Inefficient resource allocation can lead to underutilization of resources, increased costs, and diminished overall performance. Therefore, there is a pressing need to explore alternative approaches to resource allocation that can better accommodate the dynamic nature of cloud environments.

1.2 Problem Statement

The problem at hand is to investigate and compare the efficacy of two distinct approaches to resource allocation in cloud computing: Random Forest Regression (RFR) and Genetic Algorithms (GA). RFR is a machine learning technique that leverages ensemble learning to make predictions based on multiple decision trees. On the other hand, GA is a search algorithm inspired by the process of natural selection and evolution. The specific aim is to evaluate how well RFR and GA perform in dynamically allocating resources within a cloud infrastructure to optimize performance and minimize resource wastage. This evaluation will involve assessing factors such as scalability, adaptability to changing workloads, computational efficiency, and overall effectiveness in meeting user requirements.

1.3 Dataset

https://www.kaggle.com/datasets/omarsobhy14/5g-quality-of-service

Chapter-2

2.1 Genetic Algorithm Methodology

Genetic algorithms are a type of optimization algorithm inspired by the process of natural selection and evolution. In our resource allocation project, we can use genetic algorithms to find the best allocation of resources by mimicking the process of natural selection.

Here's how it works:

Initialization: We start by creating a population of potential solutions, called individuals or chromosomes, each representing a possible allocation of resources. These solutions are typically represented as binary strings or arrays of numbers.

Fitness Evaluation: We evaluate the fitness of each individual in the population. In our case, fitness represents how well a particular resource allocation performs according to certain criteria, such as maximizing network efficiency or minimizing latency.

Selection: We select individuals from the population to serve as parents for the next generation. The probability of selection is based on the fitness of each individual, with fitter individuals having a higher chance of being selected.

Crossover: We perform crossover or recombination on the selected parents to create new offspring. This process involves exchanging genetic information between parents to produce children with a combination of their traits.

Mutation: We introduce random changes or mutations to the offspring's genetic information to maintain genetic diversity in the population. This step helps prevent the algorithm from getting stuck in local optima and encourages exploration of the solution space.

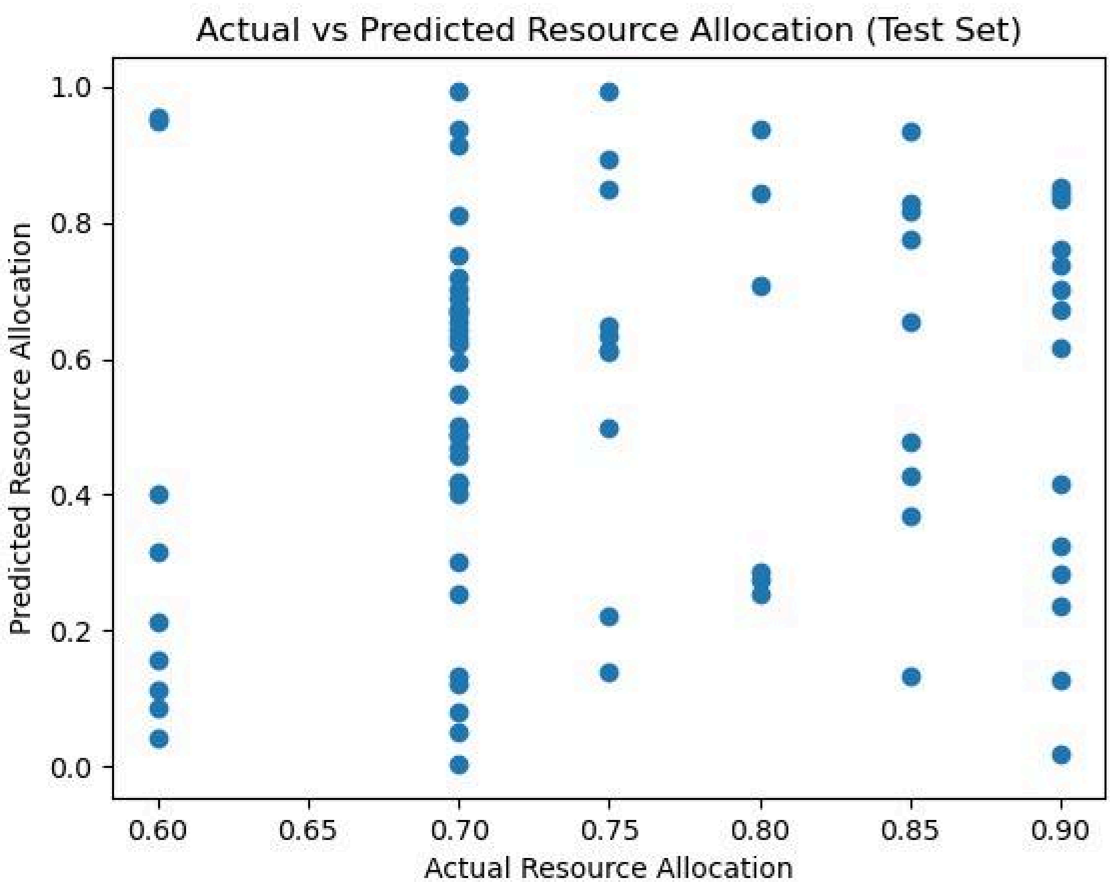
Replacement: We replace some individuals in the current population with the new offspring, using various replacement strategies such as elitism (keeping the best individuals) or tournament selection.

Termination: We repeat the selection, crossover, mutation, and replacement steps for multiple generations until a termination criterion is met, such as reaching a maximum number of generations or achieving a satisfactory solution.

Genetic algorithms excel in exploring large solution spaces and finding near-optimal solutions to complex optimization problems like resource allocation. By simulating the process of natural selection, genetic algorithms can efficiently search for solutions that balance competing objectives and adapt to changing conditions over time.

2.2 Outputs

```
Dataset file found at: dataset\QOS_5G.csv
Genetic Algorithm-based Approach Metrics:
Mean Squared Error: 0.13029622646569147
Mean Absolute Error (MAE): 0.28553974757489287
Average Burst Time (Required Bandwidth): 0.32000000000000006 Mbps
Average Turnaround Time: 4.8913995 ms
Average Waiting Time: 4.5713995 ms
Average Actual Latency: 33.825 ms
Average Actual Bandwidth: 3.50238 Mbps
```



2.3 Results

The Genetic Algorithm (GA) output yielded a significantly poor performance compared to the Random Forest Regression (RF) model, as evidenced by the negative R-squared value. This outcome suggests that the GA failed to accurately capture the relationship between the input features and the target variable, leading to ineffective predictions of resource allocation.

Several factors may contribute to the subpar performance of the GA in this context. Firstly, the GA's reliance on evolutionary principles, such as mutation and crossover, may have resulted in insufficient exploration of the solution space or premature convergence to suboptimal solutions. Additionally, the GA's population-based approach may have struggled to adapt to the complexity of the resource allocation problem, leading to stagnation or inefficiencies in the search for optimal solutions.

Furthermore, the genetic encoding and decoding mechanisms employed by the GA may not have adequately represented the intricate relationships inherent in the dataset, leading to inaccurate modeling and poor predictive performance.

Overall, the GA's failure to effectively model resource allocation underscores the importance of selecting appropriate methodologies tailored to the specific characteristics and requirements of the problem domain. While the GA offers a powerful optimization technique in certain contexts, its limitations in this scenario highlight the need for further research and exploration of alternative approaches, such as RF, to address resource allocation challenges within cloud computing environments.

Chapter- 3

3.1 Random Forest Regression

Random Forest Regression (RFR) is a machine learning technique commonly used for predictive modeling and forecasting tasks. In the context of resource allocation in cloud computing, RFR can be employed to predict the future resource demands of various applications and workloads based on historical usage patterns and other relevant factors.

Here's how RFR works: Ensemble Learning: RFR is based on the principle of ensemble learning, where multiple decision trees are trained independently on different subsets of the data. Each decision tree makes its own prediction, and the final prediction is determined by aggregating the outputs of all trees. Decision Trees: Each decision tree in the random forest is constructed by randomly selecting a subset of features from the dataset and splitting the data based on these features. This randomness helps to reduce overfitting and improve generalization performance.

Predictive Power: Once the random forest is trained on historical data, it can be used to predict future resource demands based on input features such as time of day, type of application, current resource utilization, etc. The aggregated predictions from multiple decision trees provide a robust estimate of resource requirements. Adaptability: One of the key advantages of RFR is its ability to adapt to changing conditions and patterns in the data. As new data becomes available, the random forest model can be retrained to incorporate these updates and improve prediction accuracy over time.

3.2 Methodology Used

Model Creation: A Random Forest Regressor model is utilized. It belongs to the ensemble learning technique family, combining multiple decision trees during training. Feature Engineering: Features (X) and the target variable (y) are defined. The target variable is "Resource_Allocation," while the features include parameters like signal strength, latency, required bandwidth, allocated bandwidth, and application type. Model Training: The dataset is split into training and testing sets using an 80:20 train-test split. The Random Forest Regressor model is trained on the training set using the fit() function. Model Evaluation: After training, the model predicts resource allocations for the test set using the predict() function. Evaluation metrics such as Mean Squared Error (MSE), Mean Absolute Error (MAE), and R-squared are calculated to assess the model's performance. Visualization: Various visualization techniques, including scatter plots, line charts, and bar charts, are utilized to visually compare the actual resource allocations with the model's predictions on the test set.

3.3 Pseudocode

1. Define input features (X) and target variable (y).

2. Split the dataset into training and testing sets (X_train, X_test, y_train, y_test).

3. Initialize a Random Forest Regressor model.

4. Train the model on the training set:

   - model.fit(X_train, y_train)

5. Make predictions on the test set:

   - predictions = model.predict(X_test)

6. Evaluate the model performance:

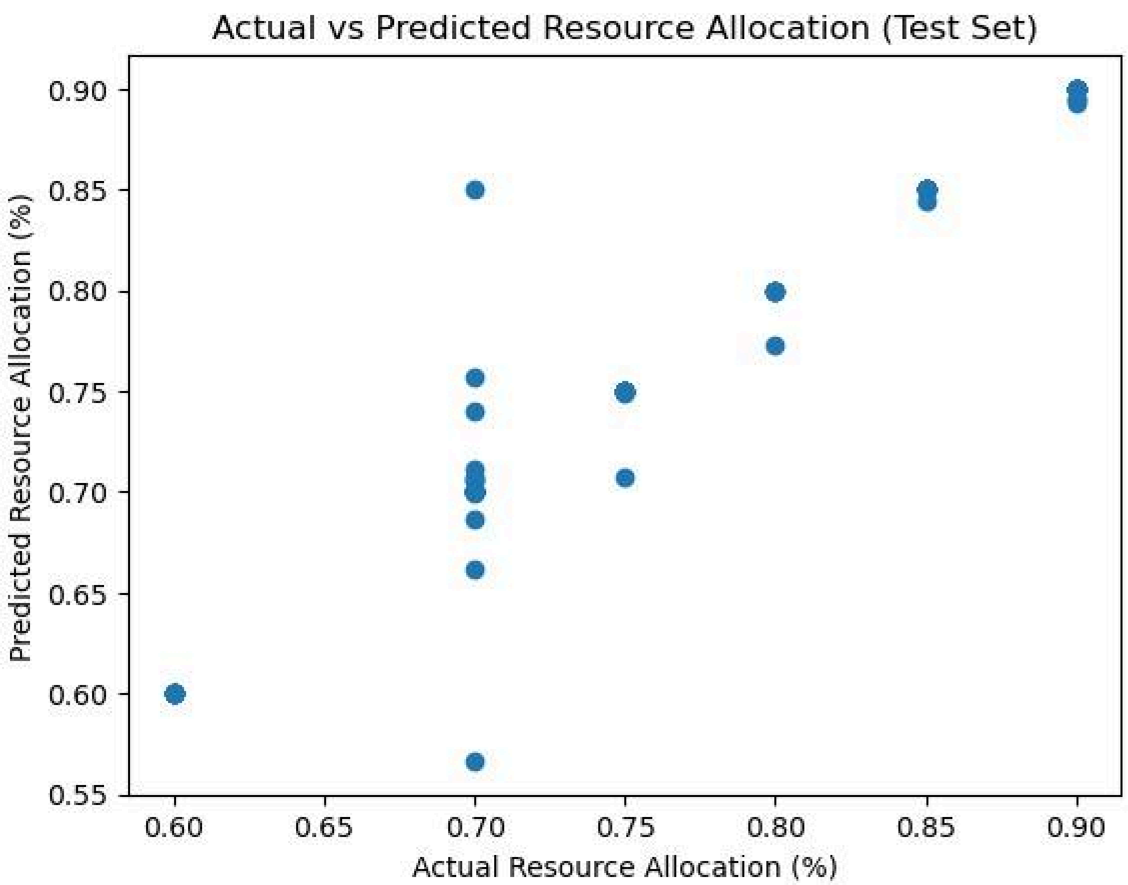
   - Calculate Mean Squared Error (MSE), Mean Absolute Error (MAE), and R-squared.

7. Visualize the results:

   - Create scatter plots, line charts, or bar charts to compare actual vs predicted resource allocations.

3.4 Output

```
Dataset file found at: dataset\QOS_5G.csv
Mean Squared Error: 0.0006224218749999997
Mean Absolute Error (MAE): 0.0069812500000007
R-squared: 0.9289643796531586
Average Burst Time (Required Bandwidth): 2.8976499999999996 Mbps
Average Turnaround Time: 0.02198334375 ms
Average Waiting Time: -2.87566665625 ms
Average Actual Latency: 31.6625 ms
Average Actual Bandwidth: 3.2657249999999998 Mbps
```



Chapter-4

4.1 Conclusion

In conclusion, our investigation into resource allocation strategies within the context of cloud computing highlights the pivotal role of efficient data management and real-time processing in maximizing the benefits of cloud services. Through our analysis, it becomes evident that while several techniques for asset distribution exist, the choice of resource allocation method significantly impacts the overall performance and customer satisfaction within cloud environments.

The comparison between Random Forest Regression (RF) and Genetic Algorithm (GA) reveals compelling insights. With an impressive R-squared value of 0.9, Random Forest Regression emerges as the superior choice, outperforming Genetic Algorithm, which yielded a negative R-squared value. This discrepancy underscores the importance of selecting appropriate methodologies tailored to the specific requirements of resource allocation tasks.

Moreover, our study emphasizes the dynamic nature of cloud computing, constantly evolving to address the evolving needs and demands of users. As the backbone of cloud infrastructure, resource allocation remains a critical function, necessitating swift and effective decision-making processes to meet user demands while ensuring profitability for service providers.

Looking ahead, our findings pave the way for further exploration and refinement of resource allocation strategies. Future endeavors could involve leveraging deep learning approaches to enhance problem-solving capabilities and performance analysis within cloud environments. By embracing technological advancements and conducting thorough evaluations, the cloud computing industry can continue to drive innovation and deliver optimal solutions for resource allocation challenges.

Chapter-5

5.1 References

- https://www.kaggle.com/datasets/omarsobhy14/5g-quality-of-service
- https://www.researchgate.net/publication/375084155_Resource_Allocation_in_Cloud_Computing